Computer Science Department

TECHNICAL REPORT

A Newton Type Algorithm
for Plastic Limit Analysis

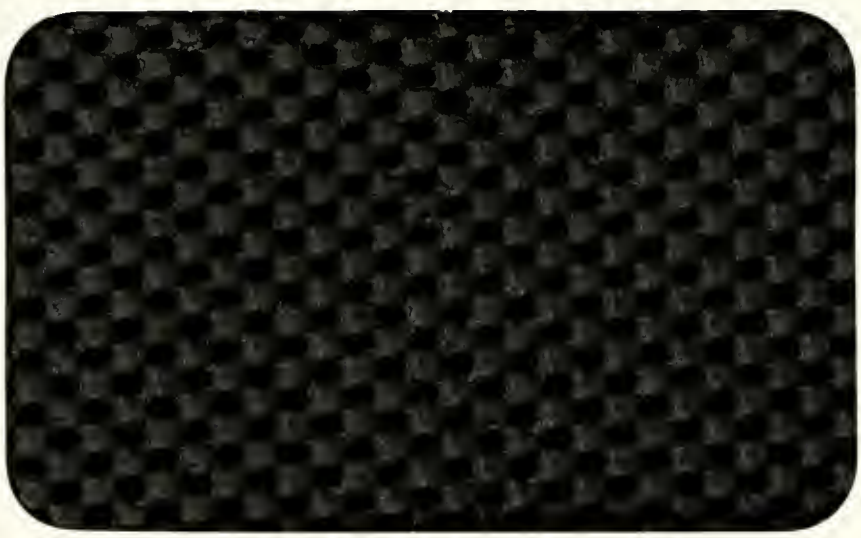
V. Gaudrat

Technical Report 422

November 1988

NEW YORK UNIVERSITY

Department of Computer Science
Courant Institute of Mathematical Sciences
251 MERCER STREET, NEW YORK, N.Y. 10012



A Newton Type Algorithm for Plastic Limit Analysis

V. Gaudrat

Technical Report 422

November 1988

A Newton Type Algorithm for Plastic Limit Analysis

Véronique F. Gaudrat†

November, 1988

† Visiting Member at the Courant Institute, supported by a French Government "Lavoisier fellowship"

A Newton Type Algorithm for Plastice Limit Analysis

VÉRONIQUE F. GAUDRAT

Courant Institute of Mathematical Sciences
New York University
New York, NY 10012
U.S.A.

Abstract. We consider a model problem from collapse load analysis for plane strain. A discretized version of the kinematic variational principle is solved by means of a Newton–type minimization algorithm. The numerical solution, which represents a possible collapse mechanism, is computed for various shapes of the domain.

I. Introduction.

This paper presents a new computational method for plastic limit analysis, based on a Newton–type minimization algorithm for the kinematic variational principle, whose solution represents a possible collapse mechanism.

The model of plastic limit analysis is by now well known, see e.g. [3, 7]. The upper and lower bound theorem [3] identifies the limit multiplier of a given load as: (i) the largest multiple of it which can be equilibrated by a statically admissible stress lying within a specified yield domain, and simultaneously as (ii) the minimum of a certain convex minimization problem among all velocity fields which do work against the load at a unit rate. The functional analytic setting of these problems has been clarified by Strang, Temam, and others [8, 9].

A common approach to the numerical solution of limit analysis problems is to approximate the yield domain by a polyhedral domain in stress space. Then the dual variational principles become a dual pair of linear programs, which may be solved, for example, using the simplex method [2]. This approach is not without disadvantages, however: the linear programming problems one arrives at are very large and highly degenerate. As a consequence, the development of improved algorithms for computing limit multipliers has been an area of continued attention, especially for problems of plane stress, plate theory, and three–dimensional structures, see e.g. [1,4].

The papers just cited all use one or another version of linear programming; in particular, they all require the replacement of a curvilinear yield domain by a polygonal approximation. A totally different approach is suggested, however by the work of M. Overton [6]. He showed that the variational problem

$$\inf_{\int_\Omega u\cdot f=1} \int_\Omega |\nabla u|, \tag{1.1}$$

can be solved quite efficiently by means of a Newton-type scheme. The success of his method is directly linked to the fact that extremals of (1.1) tend to have $\nabla u = 0$ on large portions of Ω [8]. Now (1.1) is precisely the kinematic variational principle of limit analysis for a cylinder with cross-section Ω which is loaded in antiplane shear by force per unit cross-sectional area f. Since Overton's scheme works so well in this special case, it is natural to consider using a similar method for a more difficult plasticity problem such as plane strain.

This paper presents a method analogous to Overton's for a problem of the form:

$$\inf \int_\Omega |e(u)|, \quad \Omega \subset \mathbf{R}^2, \tag{1.2}$$

subject to the constraint:

$$\int_{\Gamma_1} u \cdot f = 1, \tag{1.3}$$

where $\Gamma_1 \subset \partial\Omega$, $u = (u_1, u_2)$, f now takes values in $\mathbf{R}^2$, and $e(u) = \frac{1}{2}(\nabla u + \nabla u^t)$ is the linear elastic strain associated with u. This is the kinematic variational principle of plastic limit analysis for a two-dimensional body of shape Ω, loaded in traction along its boundary Γ_1 by force per unit arclength f, (see figure 1). The integral in (1.2) is:

$$\begin{aligned} h(u) &= \int_\Omega |e(u)| \\ &= \int_\Omega \left(\sum_{i,j=1}^{2} e_{ij}(u)^2 \right)^{\frac{1}{2}}, \end{aligned} \tag{1.4}$$

which corresponds to the yield domain

$$B = \left\{ \sigma = (\sigma_{ij}) | \sum_{i,j=1}^{2} \sigma_{ij}^2 \leq 1 \right\} \tag{1.5}$$

In other words, the dual of (1.2-1.3), which is the associated static variational principle for the plastic limit multiplier, is given by:

$$(1.6) \qquad \max\{\lambda | \exists \sigma, ||\sigma||_\infty \leq 1, \sigma \text{ symmetric }, \text{ div } \sigma = 0 \text{ in } \Omega, \sigma \cdot \vec{n} = \lambda \cdot f \text{ at } \Gamma_1\},$$

where $\vec{n}$ is the normal vector to Ω.

We should note that it is more common to use the Von Mises yield domain:

$$B' = \left\{ \sigma | \frac{1}{4} \sum_{i,j=1}^{2} ((\sigma_{ii} - \sigma_{jj})^2 + \sigma_{ij}^2) \leq 1 \right\},$$

in real applications, rather than our choice B. The use of an unbounded yield domain would complicate the implementation of our method considerably: for example, replacing B with B' would have the effect of adding the constraint div $u = 0$ in (1.2-1.3). On the other hand, it would not be difficult to adapt our method to handle an anisotropic yield condition, for example

$$B'' = \left\{ \sigma | \sum_{i,j=1}^{2} a_{ij}\sigma_{ij}^2 \leq 1 \right\},$$

with $a_{ij} > 0$ for every i, j. In any event, our (1.2-1.3) should be viewed as a model plasticity problem, effectively the first test of Overton's approach in the context of plane strain.

We consider several slightly different mechanics applications; in all of them we have a rectangular bar with two symmetric cuts of depth $(\frac{1-l}{2})$ in the y–direction. The bar is loaded in tension along opposite sides parallel to the cuts (see figure 1). In the different applications the shape of the cuts changes.

The Newton method works rather well. In most cases the solution has $|e(u)| = 0$ on most of Ω, in other words, the collapse mechanism consists of a dislocation along the curve which joins the two cuts, with the part of Ω on either side of the dislocation moving as a rigid body. Though we know of no theoretical reason for the collapse mechanism to have this form, it is certainly plausible. No doubt the tendency to have such a collapse mechanism is linked to the success of the algorithm.

The paper will be organized as follows: in the next part the discretization of the problem introduced in [1] (with infinitesimally thin cuts) will be presented. The method and the

algorithm are given in part III. Part IV discusses the implementation of the algorithm with the construction of the descent direction and the method used to test different shapes of the domain with the same program. In the last part the numerical results are given for different shapes of the cuts.

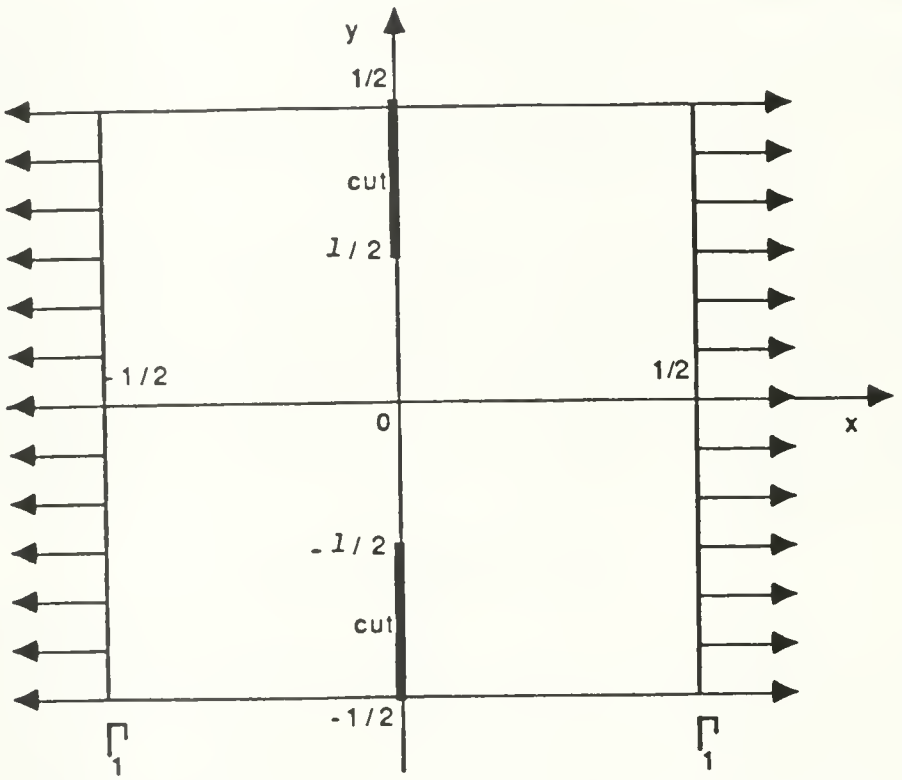


figure 1: the domain Ω where the cuts are infinitesimally thin.

II Discretization.

Using the symmetry of the domain and forces the problem can be solved on the quadrant $0 \leq x \leq \frac{1}{2}, 0 \leq y \leq \frac{1}{2}$. We will still call this part of the domain Ω. The constraints arising from the symmetry are:

$$\begin{cases} u_1(0,y) = 0 & 0 \leq y \leq l/2, \\ u_2(x,0) = 0 & 0 \leq x \leq \frac{1}{2}. \end{cases} \tag{2.1}$$

We follow the notational conventions of Overton [6]. Let N be the number of mesh points in each direction and $\beta = 1/N$ the mesh size. The mesh point corresponding to the bottom of the cut ($y = l/2$) will be indexed by l (see figure 2). The size of the problem

(1.2-1.3) is $2N^2$. The function u in (1.2-1.3) is discretized using piecewise linear finite elements: its value at the $(i,j)^{\text{th}}$ mesh point is $(u_1(i,j), u_2(i,j))$.

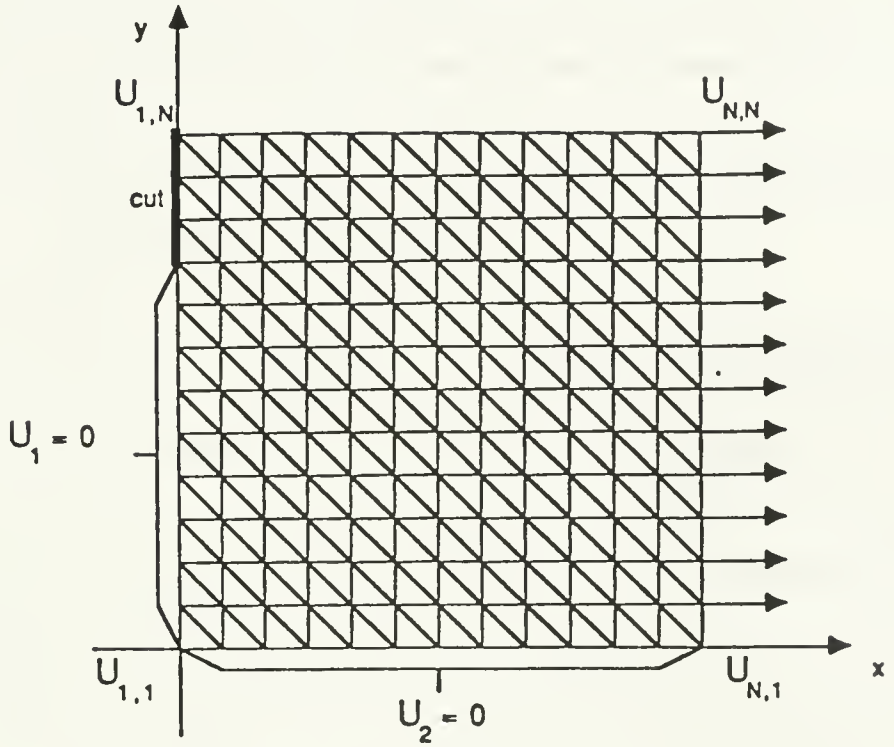


figure 2: discretization of the domain.

The discrete strain is noted e_{ijp} where $i,j = 1,\ldots N-1$ and $p = 1,2$ corresponds to the lower or upper triangle; thus on a lower triangle we have (see figure 3):

$$e_{ij1} = \beta \begin{bmatrix} u_1(i+1,j) - u_1(i,j) & A \\ A & u_2(i,j+1) - u_2(i,j) \end{bmatrix}, \tag{2.2}$$

where

$$A = \frac{1}{2}(u_1(i,j+1) - u_1(i,j) + u_2(i+1,j) - u_2(i,j))$$

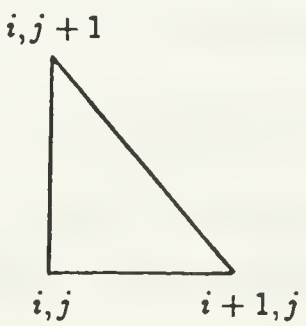


figure 3: indices on a lower triangle

A change of indices gives the discrete strain on the upper triangle, e_{ij2}. During the computation only three terms of the matrix of the strain need to be stored; the fourth can be deduced by symmetry.

From (1.4) and (2.2) we deduce the discrete norm of the strain:

$$|e_{ij1}| = \left[\frac{(u_1(i+1,j) - u_1(i,j))^2}{\beta^2} + \frac{(u_2(i,j+1) - u_2(i,j))^2}{\beta^2} \right.$$

$$\left. + \frac{1}{2} \frac{(u_1(i,j+1) - u_1(i,j) + u_2(i+1,j) - u_2(i,j))^2}{\beta^2} \right]^{\frac{1}{2}}; \tag{2.3}$$

the discretized version of (1.2) is thus:

$$\min_{u(i,j)} \sum_{i,j,p} \frac{\beta}{2} |e_{ijp}|, \quad i,j = 1, \ldots, N \text{ and } p = 1,2, \tag{2.4}$$

subject to the constraint coming from the symmetry (2.1),

$$\begin{cases} u_1(1,j) &= 0 \quad 1 \le j \le j_l, \\ u_2(i,1) &= 0 \quad 1 \le i \le N, \end{cases} \tag{2.5}$$

and the constraint (1.3) :

$$\beta \sum_j f_{Nj} u_1(N,j) = 1. \tag{2.6}$$

Although (2.4), (2.5) and (2.6) is a finite dimensional convex optimization problem, it is not easily solved because the objective function to be minimized is not differentiable with respect to u if any of the residuals $|e_{ijp}|$ is equal to zero. Note that the residual vanishes on a triangle when the displacement there corresponds to an infinitesimal rigid body motion. In the numerical discussion we will call this situation a "zero–residual."

A classical way to solve a problem with a non–differentiable function is to consider the continuously differentiable approximation of h, for example:

$$h_\epsilon(u(i,j)) = \sum_{i,j,p} \beta (e_{ijp}^2 + \epsilon_\beta)^{\frac{1}{2}}, \epsilon_\beta > 0. \tag{2.7}$$

Notice that h_ϵ is differentiable everywhere provided that $\epsilon_\beta > 0$ (we shall actually let ϵ_β depend on the mesh size β). Unfortunately, for a small value of ϵ_β the problem is very

ill–conditioned and for a large value of ϵ_β the solution we reach using this approximation is too far from the solution of (1.2-1.3).

The algorithm presented below works directly with h, not with h_ϵ. However, it uses h_ϵ to find an initial displacement and to attempt to verify optimality of the final point, as will be explained.

In the following part we will introduce the method we use to solve this problem.

III. The method

Overton proposed in [5] an algorithm for minimizing functions of the form : sum of the square roots of sums of squares. The idea of the algorithm is to restrict the objective function to the linear subspace where the zero–residuals remain unchanged. In this space, the objective function is continuously differentiable, and so it may be minimized using Newton's method. Then of course one must check whether the "solution" so obtained is also optimal in the larger space. As presented in [5], the algorithm requires a "full rank condition" which is not valid for the plastic limit analysis functional. However, Overton adapted the algorithm in [6] to solve (1.1). He gave a simple way to identify the subspace where zero–residuals remain unchanged and to construct the restricted objective function.

The algorithm is organized as follows : first, define the subspace where zero–residuals remain unchanged ; then update the restricted objective function, its gradient and its hessian ; and finally, check for optimality. Before going into the details of the algorithm and its implementation we will give some definitions and introduce the restricted objective function. Then a way to check the optimality will be presented.

III.1. Definition of the subspace and the restricted objective function

We will use the notations of Overton in [6]. In the following the index j will represent $(i,j,p), i,j = 1,\ldots,N-1$ and $p = 1,2$. The objective function (1.4) is written

$$h(u) = \sum_{j=1}^{M} \beta |e_j(u)|, \tag{3.1}$$

where $u \in \mathrm{R}^{2N^2}$, $M = 2\cdot(N-1)^2$ and $e_j(u) = B_j u, where \epsilon B_j$ is a $(4 \times 2N^2)$ matrix.

The active set at the point u is the set of triangles with zero–residuals :

$$J(u) = \{j \,||e_j(u)| = 0\}\,, \tag{3.2}$$

and the active terms or residuals are $e_j(u)$ such that $j \in J(u)$. Let $\hat{B}$ be the matrix whose columns are the coefficient matrix of the active residuals:

$$\hat{B} = \left[B_{j_1}, B_{j_2}, \ldots, B_{j_p}\right], \text{ where } J(u) = \{j_1, j_2, \ldots, j_p\}\,, \tag{3.3}$$

and let $Z = Z(u)$, the matrix whose columns span the null space of $\hat{B}^t$. The matrix Z satisfies :

$$\hat{B}^t Z = 0, \tag{3.4}$$

and

$$e_j(u + Zp) = e_j(u) = 0, \quad \forall j \in J(u), \quad \forall p \in \mathbf{R}^{2N^2} \tag{3.5}$$

Now for any u we can define the objective function h restricted to the manifold defined by $u \oplus R(Z)$ as:

$$h_z(p_z) = h(u + Zp_z). \tag{3.6}$$

Consider the gradient and the hessian of the inactive terms:

$$g_j(u) = \nabla|e_j(u)| = \frac{B_j e_j(u)}{|e_j(u)|}, \forall j \notin J(u), \tag{3.7}$$

$$G_j(u) = \nabla^2|e_j(u)| = \frac{B_j B_j^t}{|e_j(u)|} - \frac{B_j e_j(u) e_j^t(u) B_j^t}{|e_j(u)|^3}, \forall j \notin J(u) \tag{3.8}$$

then the gradient and the hessian of h_z are given by:

$$\nabla h_z(p_z) = Z^t g(u + Zp_z), \quad \text{where } g = \sum_{j \notin J(u)} g_j, \tag{3.9}$$

$$\nabla^2 h_z(p_z) = Z^t G(u + Zp_z) Z, \quad \text{where } G = \sum_{j \notin J(u)} G_j. \tag{3.10}$$

III.2. Testing the optimality of the subspace

Once a displacement is found which is optimal in the restricted space, we must check whether it is also optimal for the entire (unrestricted) space. Because the full rank condition mentioned earlier does not hold, this is difficult. Following Overton [6], we only check if we can find a lower solution in the unrestricted space, using the continuously differentiable approximation of h (2.7). We perform few iterations of this algorithm with the "solution" in the restricted space as initial point. If a lower point is found, we perform another optimization using different parameters to identify the active set (see part V for details). Although this technique is not guaranteed to work, Overton [6] gives some arguments explaining why the algorithm is unlikely to converge to a nonoptimal point.

III.3. The algorithm

The main steps which must be performed to obtain a new iterate u^{k+1} from u^k are:

1. *Choose an initial point* u^0 for which the hessian is non–singular.
2. *Identify the active set.*
   - 2.1 Merge into the active set all the triangles which have an exact zero–residual.
   - 2.2 Merge into the active set all the elements with a very small residual provided that in doing so the objective function decreases.
   - 2.3 Only if the objective function does not increase significantly, merge into the active set all the elements with small residuals and those surrounded by zero or almost zero–residual triangle
   - 2.4 If the merging in (3.2) only is effective go to (3.1) to check if there are new zero–residuals to merge.
   - 2.5 Update Z.
3. *Perform an optimization step in the space* $u \oplus R(Z(u))$.
   - 3.1 Compute a descent direction using the restricted gradient and hessian with Newton's method if the restricted hessian is well conditioned, otherwise with the steepest descent method.
   - 3.2 Perform a line search in that direction.

4. *Termination tests.*

Stop if the number of iterations is too large.

Check the optimality of the subspace using the function h_ϵ if the restricted gradient is sufficiently reduced.

IV Implementation.

In this part we will give some details of the implementation of the various steps of the algorithm.

IV.1. Choosing the initial point.

The problem is so badly conditioned that the initial point must be chosen very carefully. Since we only augment the active set until we find a minimum in the subspace, the initial point should not have any of its residuals equal to zero. To find an initial point which satisfies this condition we perform a few iterations of an approximate continuously differentiable problem using the function h_ϵ defined in (2.7) with ϵ_β set equal to $10^{-1}\beta$. This method has the advantage of giving a point close to a solution with all residuals strictly positive.

IV.2. Identifying and updating the active set.

When an element is added to the active set, i.e. when the value of the corresponding residual is small, the matrix Z and the value of u must be updated. We now consider the relation between elements of the active set, in order to be able to update the matrix Z efficiently.

A zero–residual element satisifies:

$$|e_j(u)| = 0,$$

which is equivalent, for a lower triangle, to:

$$(4.1) \quad \begin{cases} \beta/\sqrt{2} & (u_1(i,j-1) - u_1(i-1,j-1)) = 0, \\ \beta/\sqrt{2} & (u_2(i-1,j) - u_2(i-1,j-1)) = 0, \\ \beta/\sqrt{2} & (u_1(i-1,j) - u_1(i-1,j-1) + u_2(i,j-1) - u_2(i-1,j-1)) = 0, \end{cases}$$

and for an upper triangle:

$$(4.2)\qquad \begin{cases} \beta/\sqrt{2} & (u_1(i,j) - u_1(i-1,j)) = 0, \\ \beta/\sqrt{2} & (u_2(i,j) - u_2(i,j-1)) = 0, \\ \beta/\sqrt{2} & (u_1(i,j) - u_1(i,j-1) + u_2(i,j) - u_2(i-1,j)) = 0. \end{cases}$$

Here we see the biggest difference between our problem and Overton's : a zero–residual element is determined by three parameters (an infinitesimal rigid motion in $\mathbf{R}^2$) instead of just one as in [6]. Let us write:

$$(4.3)\qquad \begin{cases} k_1 & = u_1(i,j-1) = u_1(i-1,j-1), \\ k_2 & = u_2(i-1,j) = u_2(i-1,j-1), \\ \Delta & = u_1(i-1,j) - u_1(i-1,j-1) = -u_2(i,j-1) + u_2(i-1,j-1). \end{cases}$$

A problem arises which has no analogue in Overton's algorithm: how to define the parameters (4.3) for a triangle newly merged with an active set, if this triangle meets another triangle of the active set.

If the two triangles share an edge then the resolution of this problem is easy: the three parameters are uniquely defined by the relations (4.3). We will call a zero–residual zone such a subset of $J(u)$. It defines the motion of a rigid part of the domain. Thus on a zero–residual zone, u has a constant value on each line parallel to the x–axis and a constant slope Δ in the y direction. Symmetrically u_2 has a constant value in the y direction and a slope $-\Delta$ in the x direction. The displacement on the zero residual zone can be written

$$(4.4)\qquad \begin{cases} u_1(i,j) & = & k_1 + \Delta(j - j_0), \\ u_2(i,j) & = & k_2 - \Delta(i - i_0), \end{cases}$$

where k_1, k_2 are chosen such that $u_1(i_0,j_0) = k_1$ and $u_2(i_0,j_0) = k_2$.

If, however, the two triangles share only one mesh point then the resolution of the "problem" is not so clear. In principle the slope Δ could be different on each zone.However, we expect that in most cases (near the solution) the value of Δ will be the *same* on all the triangles linked by a mesh point. A physical explanation of this assertion is as follows : Consider two rigid bodies Ω_1 and Ω_2 linked by a point P_0 (see figure 5). The energy is null on $\Omega_1 \cup \Omega_2$ although we know from (4.3) that Δ can be different on Ω_1 and Ω_2. But then consider a convex domain Ω such that $(\Omega_1 \cup \Omega_2) \subset \Omega$: the energy on the domain Ω depends strongly on the value of the two parameters, i.e. if the displacement is constant on

$\Omega \setminus \Omega_1 \cup \Omega_2$ but the value of Δ is different on Ω_1 and Ω_2, then the global energy ($\int |e(u)| d\Omega$) cannot be too small. We thus make the convention that if two zero–residual zones meet as in figure 4, then we shall combine them into a single zero–residual zone (and hence require that u be the same infinitesimal rigid motion on each). A numerical consequence is that all mesh points of the new zero–residual zone will satisfy the relations (4.4) with the same value of the parameters.

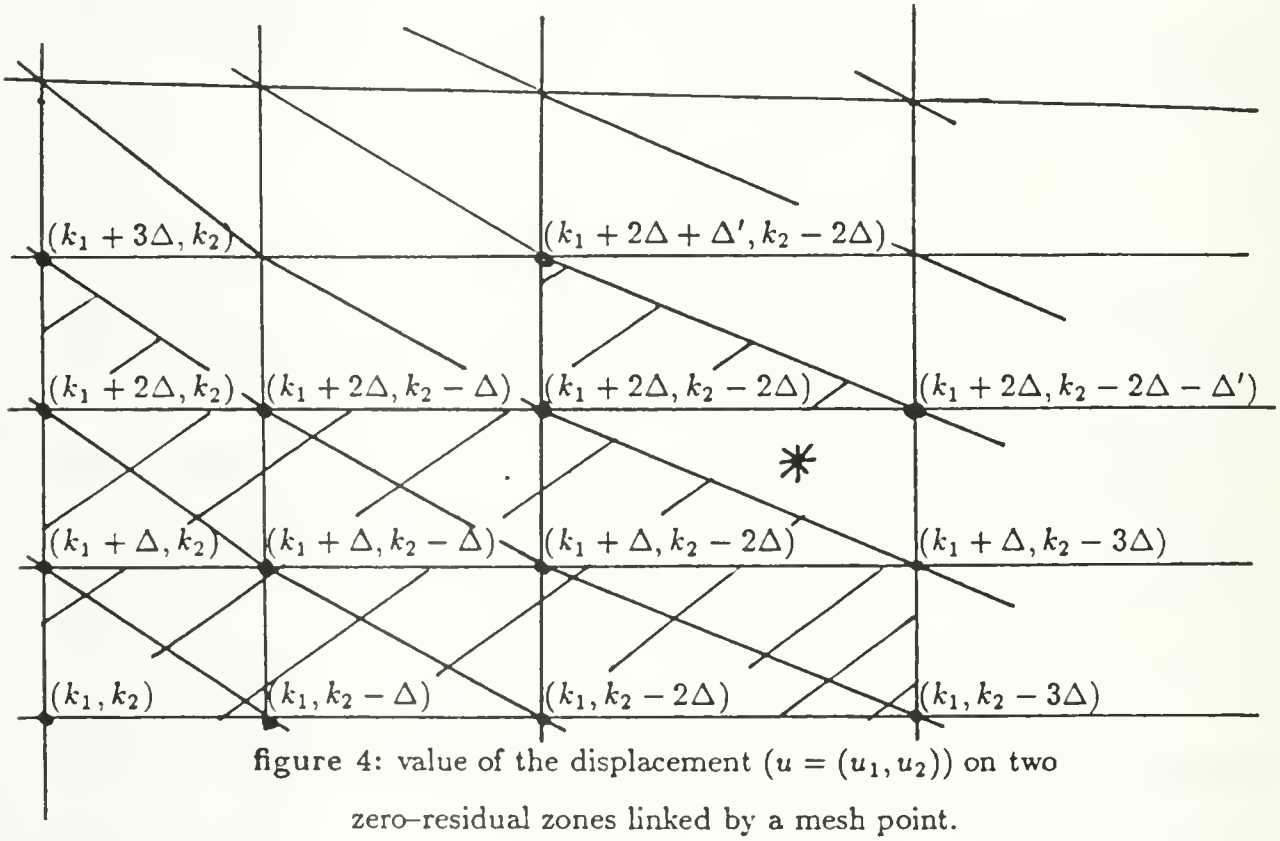


figure 4: value of the displacement ($u = (u_1, u_2)$) on two zero–residual zones linked by a mesh point.

Thus a triangle such as one maked $*$ in figure 4 should satisfy the relation (4.3) and belong to the zero–residual zone. As a result of this convention the global energy may increase after the merging of zero-residual zones. In the following, by a "forced merging", we mean a merging which is performed although the global energy increases.

To insure the global convergence of the algorithm new triangles are merged with the active set only if the value of the objective function does not increase too much as a result, and if we are close to a zero–residual, i.e. if their residuals are very small or if the condition number of the hessian is very large. The value of the displacement is stored if it corresponds to the lowest energy ever computed by the algorithm. If the termination

test is positive, i.e. if the gradient is sufficently small, but the objective function is not the lowest ever computed, we must perform another optimization from the stored value of the displacement with different parameters for the merging operation. (For the details see the numerical part).

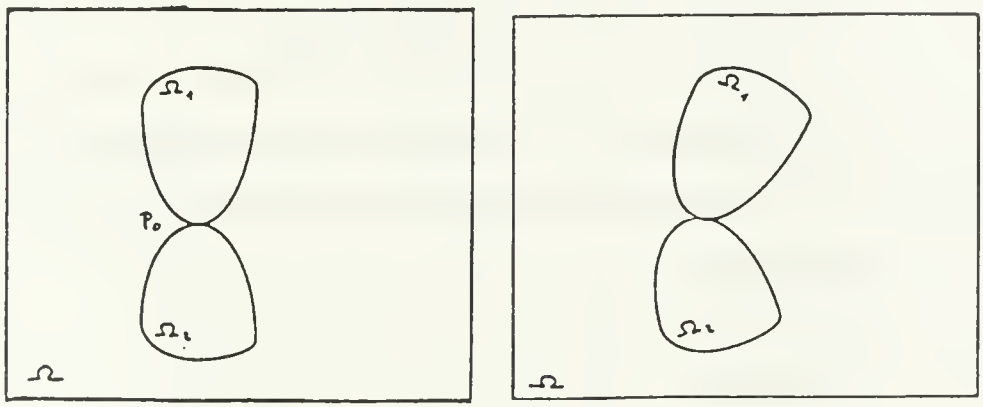


figure 5. position of the two rigid bodies Ω_1 and Ω_2 in Ω for different values of the parameters Δ.

IV.3. Computing the restricted objective function.

We consider now the representation of the displacement on Ω. Our goal is to deduce the form of the matrices $\hat{B}$ and Z introduced in III.1.

The set of mesh points is divided into lists such that every element belongs to one and only one list. All mesh points in Ω which belong to the same zero–residual zone will be in the same list and a mesh point which does not belong to any zero–residual will be considered as a list of one element.

The displacement on a list is determined by only two parameters (k_1, k_2) if it is a list of one element and the three parameters k_1, k_2 and Δ defined in (4.3) if the list has more than one element. The first element of the list, indexed by (i_0, j_0), is such that $u_1(i_0, j_0) = k_1$, $u_2(i_0, j_0) = k_2$.

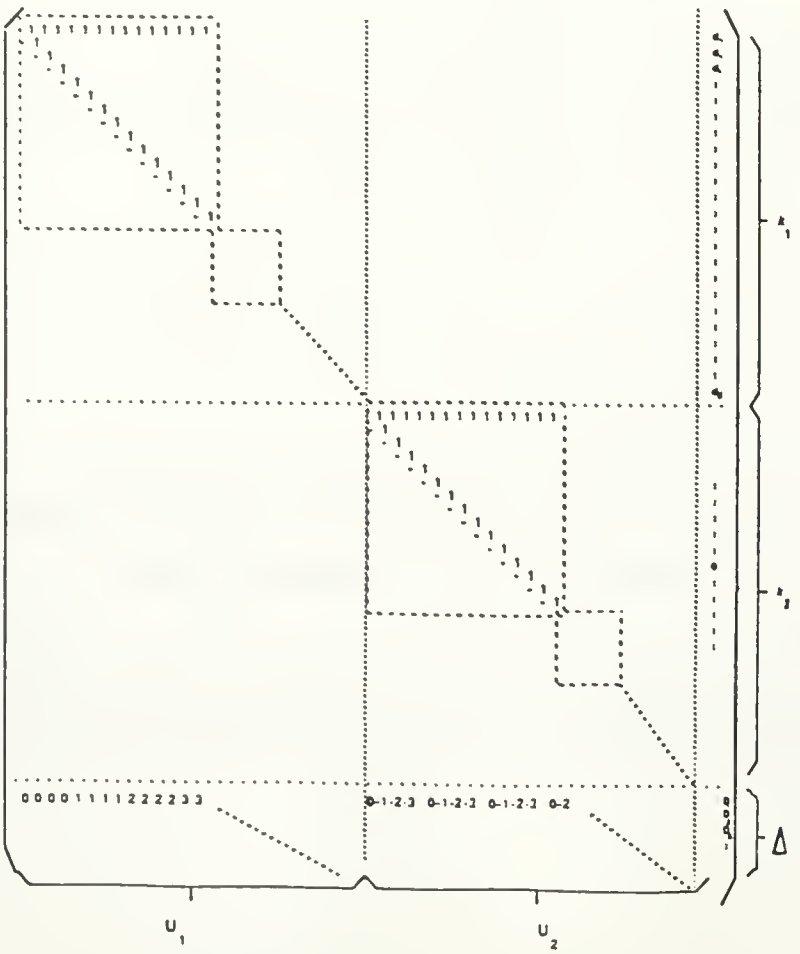
The constraint arising from the symmetry (2.1) can now be expressed in the following way: If the displacement in the x–direction or the y–direction of one element of a list is

fixed to zero by the constraint (2.1), then the corresponding parameter (k_1 or k_2) will be fixed to zero and the given element will be the first of the list. (Note that if there are two or more elements with a fixed displacement, the parameter Δ has to be zero and it does not matter which one is chosen as the first one). If the displacement in the x-direction and/or the y-direction of two or more elements are fixed to zero, then the value of the corresponding parameter(s) k_1, k_2 and Δ are fixed to zero. If the value of one or more parameters of a list is fixed to zero the list is called a "fixed list."

Using this structure and the relation (4.4) we can deduce a simple form for the matrix $\hat{B}$ defined in (3.3). The matrix $\hat{B}$ can be divided into two column blocks corresponding to u_1 and u_2, and three row blocks corresponding to the three parameters k_1, k_2 and Δ.

The size of the columns block is fixed but the size of the three row blocks depends on the number of parameters defined and not fixed to zero. If we call n_1, n_2, n_3 respectively the size of the three row blocks, the size of the matrix is $((n_1 + n_2 + n_3) \times 2N^2)$.

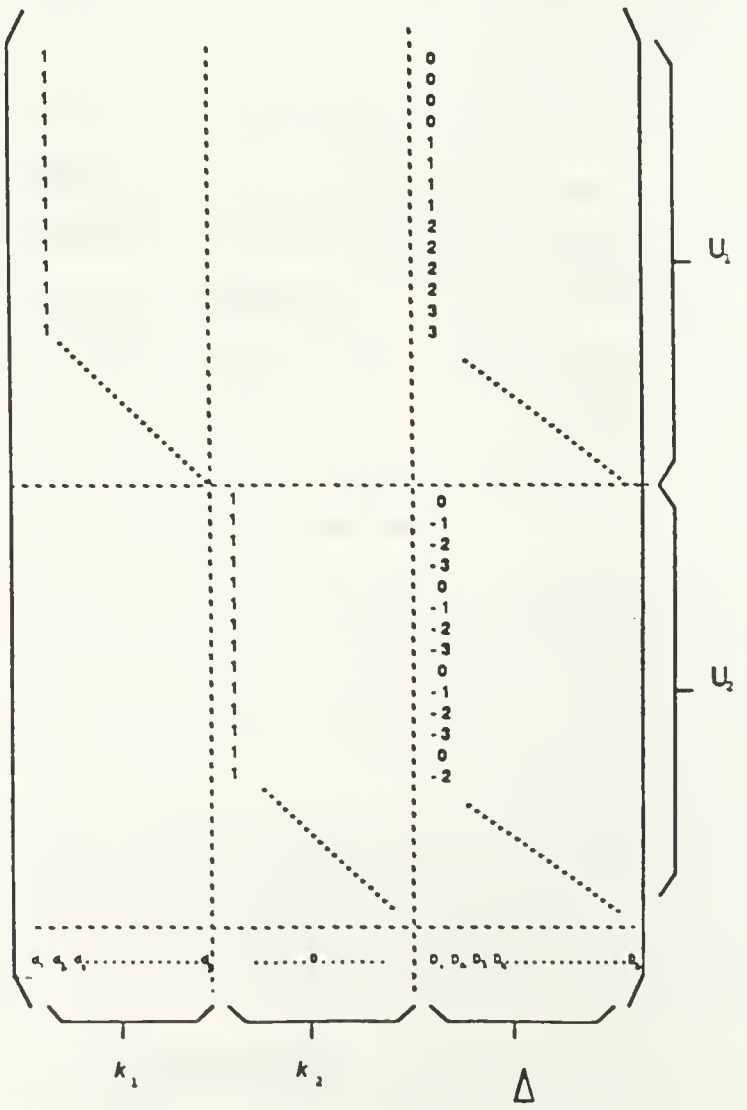
Taking into account that the constraint (2.6) must be satisfied in any direction of search p, we can write $\hat{B}$ in the following way where the first list coresponds to the list represented in figure 4 with $\Delta = \Delta'$:



The last column is defined by the relation on the list s:

$$\begin{aligned}\sum_{(i,j)\in \text{ list } s} f(i,j)u_1(i,j) &= \sum_{(i,j)\in \text{ list } s} f(i,j)(u_1(i_0,j_0)+(j-j_0)\Delta_s)\\ &= k_1\left(\sum_{(i,j)\in \text{ list } s} f(i,j)\right)+\Delta_s\left(\sum_{(i,j)\in \text{ list } s}(j-j_0)f(i,j)\right)\\ &= k_1 d_s - \Delta_s D_s. \qquad (4.5)\end{aligned}$$

Storing and factoring $\hat{B}$ would be prohibitively expensive. Fortunately, a full rank matrix Z which spans $N(\hat{B}^t)$ can be written in the following form:



The matrix Z is divided in blocks conformed with those of $\hat{B}^t$. The last row of the matrix insures that the constraint will be respected by any search direction. To insure

the numerical stability, the coefficients of the last row defined in (4.5), are scaled in the following way:

$$d'_s = -d_s/d_t \quad \text{and} \quad D'_s = -D_s/d_t,$$

$$\text{with } t \text{ such that} \quad |d_t| = \max\{|d_1|, \ldots, |d_s|\}.$$

Notice, there are no rows corresponding to u_1 for the list "t"; if this list has more than one element, then the last row will be repeated as many times as necessary.

Remark : This technique can easily be extended to the case where u_1 and u_2 are constrained over the whole domain by adding the corresponding terms in the last row. Only the lists and their structure are stored ; thus the updating operation of $\hat{B}$ and Z is limited to the updating of the list.

IV.4. The optimization in the restricted space.

As for the problem of minimizing the energy in the space $u \oplus R(z)$, we first factor the matrix Z^tGZ and estimate its condition number using the Gauss elimination method as implemented in LINPACK. If the condition number is not too large, we then solve the system of equations. If the condition number is greater than 10^{10} we simply take the negative gradient to be the surch direction. We cannot use the conjugate gradient method to solve the restricted Newton system because the numerical errors are too large; thus we must store the whole matrix Z^tGZ and increase the number of computations per iteration. Note that the restricted hessian is sparse, therefore an algorithm that takes advantage of this could be used to speed up the method.

The line search is exactly the same as in Overton [5].

IV.5. Modifying the domain.

The structure of the algorithm actually makes it easy to handle changes in the shape of the domain with only a minimal change in the program. Consider a domain $\Omega' \subset \Omega$ with a regular boundary. We set the energy outside of Ω' to zero by setting:

$$h_{\Omega'}(u) = \beta \sum_{j=1}^{M} \delta_j |e_j(u)|,$$

$$\text{where} \quad \delta_j = \begin{cases} 0 & \text{if} \quad j \notin \Omega', \\ 1 & \text{if} \quad j \in \Omega'. \end{cases}$$

In the next part, we will present numerical results for different shapes of the domain obtained by using this technique.

V. Numerical results.

The fortran program used to obtain the results was written by modifying and extending the program of Overton used to obtain the results reported in [6]. It was run on a Vax 11/780 at the Courant Institute Mathematics and Computing Laboratory. Double precision arithmetic, i.e. approximately 16 decimal digits of accuracy, was used throughout.

We tried several problems including the specific one above. We considered different shapes of the cuts: infinitesimal by thin cuts as in [1] (figure 6) ; large rectangular cuts, two mesh sizes wide (figure 7) ; and triangular cuts (figure 8). We studied each of these problems for three different lengths of the cuts: $l = \frac{1}{3}, \frac{1}{2}, \frac{2}{3}$.

To keep the symmetry of the continuous problem we have used a symmetric discretization. Thus for the first and second shape we have solved the discrete problems for two different discretizations (see figure 9). In the third problem only one discretization is possible. In all the tests presented here, we consider only the constant load (1.3):

$$f(x,y) = \begin{cases} 1 & x = \frac{1}{2}, -\frac{1}{2} \leq y \leq \frac{1}{2}, \\ 0 & \text{elsewhere.} \end{cases} \tag{5.1}$$

Computational experience indicates that the specific solution we reach and the speed of the method depends strongly on the value of the parameters used in step 2 of the algorithm: ϵ_{MCH} which determines when a residual is considered exactly active (i.e. to be a zero-residual), ϵ_{ACT} which determines when an attempt is made to make an almost active residual exactly active, ϵ_{COND} which determines if the condition number is sufficiently large to authorize a "forced merging" defined in IV.2, and ϵ_{FOR} which is used when the merging is forced (the smaller residual, while non-zero, must be smaller than $\epsilon_{\text{FOR}} * \epsilon_{\text{ACT}}$).

If the termination test is successful but the value of the objective function is not the smallest computed, the algorithm starts again with smaller values for the parameters ϵ_{FOR}, ϵ_{ACT}, ϵ_{MCH} and a larger value for the parameter ϵ_{COND}, with the computed "solution" as

an initial point. Note that a merging can be "forced", i.e. the objective function increased after the merging, although the residual was "exactly active", i.e. $|e_i| < \epsilon_{\text{MCH}}$.

The parameters are updated in the following way:

$$\begin{aligned} \epsilon_{\text{MCH}} &= \epsilon_{\text{MCH}} \times 10^{-2}, \\ \epsilon_{\text{ACT}} &= \epsilon_{\text{ACT}} \times 10^{-1}, \\ \epsilon_{\text{FOR}} &= \epsilon_{\text{FOR}} \times 10^{-1}, \\ \epsilon_{\text{COND}} &= \epsilon_{\text{COND}} \times 10. \end{aligned}$$

The usual value of the condition number is 10^6; a "large" value is typically 10^{22}. If it is larger than 10^{10} a steepest descent step is performed instead of a Newton step.

The algorithm is not very sensitive to the value of ϵ_{PG}, which determines if the gradient is sufficiently reduced; it has been fixed to 10^{-6}. The parameters ϵ_{LINE} and ETA used in the line search are the same as in Overton's problem [5, 6].

The tables I and II summarize the results for different problems and two different initial values of the parameters $\epsilon_{\text{ACT}}, \epsilon_{\text{MCH}}$ and ϵ_{PG}. In the last column of the tables is given an estimation of the speed of the algorithm:

$$\text{EST} = \frac{\sum_{i=1}^{Q} mi^3}{Qm_0^3}, \tag{5.2}$$

where mi is the size of the restricted space at the ith iteration, m_0 is the size of the initial problem and Q the number of iterations. This estimator does not take into account the number of iterations needed and the iteration where the steepest descent method is used. In all tests performed the number of mesh points N is 12.

Physical intuition suggests that the solution in all cases should be:

$$u_1(x,y) = \begin{cases} \bar{u}_1 & 0 < x \le \frac{1}{2}, \quad -\frac{1}{2} \le y \le \frac{1}{2}, \\ 0 & x = 0, \quad -\frac{1}{2} \le y \le \frac{1}{2}, \\ -\bar{u}_1 & -\frac{1}{2} \le x < 0, \quad -\frac{1}{2} \le y \le \frac{1}{2}, \end{cases} \tag{5.3}$$

$$u_2(x,y) = 0 \quad -\frac{1}{2} \le x \le \frac{1}{2}, \quad -\frac{1}{2} \le y \le \frac{1}{2}.$$

The solutions we found are very close to this intuitive solution. Note that the form of the solution is part of the success of the algorithm ; among other solutions, this algorithm always seems to choose this type of highly discontinuous solution.

We represent the solutions (figures 10, 11 and 12) with the same method as used by Christiansen in [1]: the position (d_1, d_2) of a mesh point for the optimal displacement is obtained by:

$$\begin{aligned} d_1(x_i, y_j) = &\quad x_i + \alpha u_1(x_i, y_j) \\ d_2(x_i, y_j) = &\quad y_i + \alpha u_2(x_i, y_j) \end{aligned}$$

Our graphics are obtained by using $\alpha = 5/6$. In figure 10 the slight difference from the intuitive solution is mainly due to the discretization. In figure 13 we represent the value of the displacement after 13 iterations of a Newton method applied to the approximated problem (2.7), using shape 1, discretisation 1 with the parameter values $l = \frac{1}{2}, \epsilon = 0.1\beta$. The value of the projected gradient is $3.9 \cdot 10^{-13}$ and the value of the objective function is 0.944. Note that the value of the objective function of the same problem in table I is 0.719.

Conclusion.

In this paper we have described a method to solve the problem (2.4), a discretization of (1.2-1.3). Although the problem is more complex than M. Overton's problem [6], the method handles the ill–conditioning of the hessian very well, and allows one rather quickly to find a better solution than that obtained using a continuously differentiable approximation of the objective function (2.7). The success of the algorithm is presumably linked to the structure of the solution (5.3), which, though highly intuitive, has not yet been proved (except numerically) to be optimal.

Acknowledgements.

I would like to thank the Courant Institute for its warm hospitality and especially Professors R. Kohn and M. Overton for suggesting the subject of this paper, for their guidance and for their kindness throughout my stay.

References

[1] Christiansen E., *Computation of Limit Loads*, Int. Jour. Num. Meth. Eng. **17** (1981), 1547–1570.

[2] Dorn W. S., Greenberg H.J., *Linear Programming and Plastic limit Analysis of Structures*, Quaterly of Applied Mathematics **15** (1957), 155–.

[3] Hodge P.G., "Plastic Analysis of Structures," McGraw-Hill, 1959.

[4] Kortanek K.O., *Semi Infinite Programming and Continuum Physics*, Lecture Notes in Economics and Mathematical Systems 259, Anderson E.J., Philott A.B. eds., Springer Verlag (1985), 65–78.

[5] Overton M.L., *A Quadratically Convergent Method for Minimizing a sum of Euclidean Norms*, Math. Programming **27** (1983), 34–63.

[6] Overton M.L., *Numerical Solution of a Model Problem from Collapse Load Analysis.*, in "INRIA Conference : Comp. Meth. in Eng. and Appl. Sciences," Glowinski R., Lions J.L. eds., 1984, pp. 421–437.

[7] Prager W., Hodge P.G., "Theory of Perfectly Plastic Solids," John Wiley and Sons, (1951).

[8] Strang G., *A Minimax Problem in Plasticity*, Functional Analysis and Numerical Analysis, Nashed M. Z. ed., Lecture Notes in Mathematics 701, Springer Verlag (1979), 319–333.

[9] Temam R., "Problemes Mathématiques en Plasticité," Gauthier-Villars, Paris, 1983.

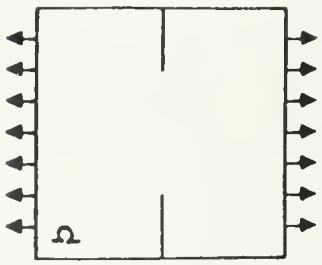


fig6:infinitesimal cuts(shape 1)

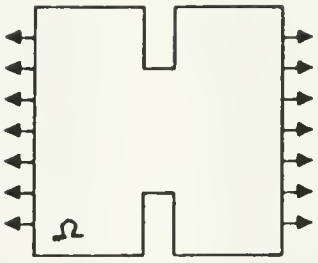


fig7:rectangular cuts(shape 2)

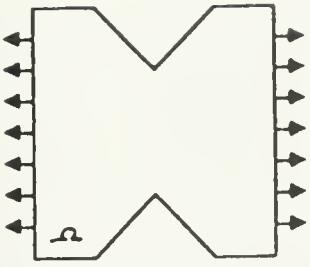
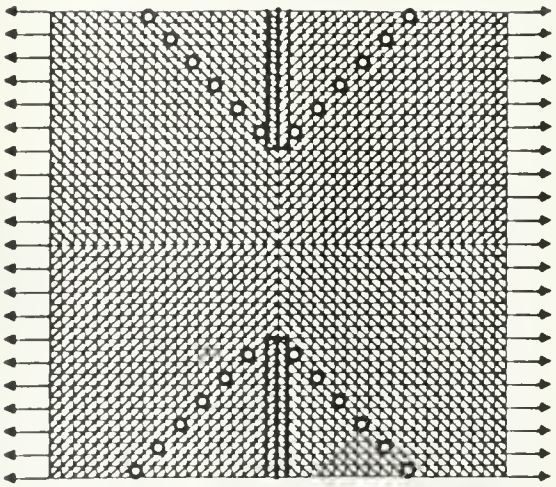
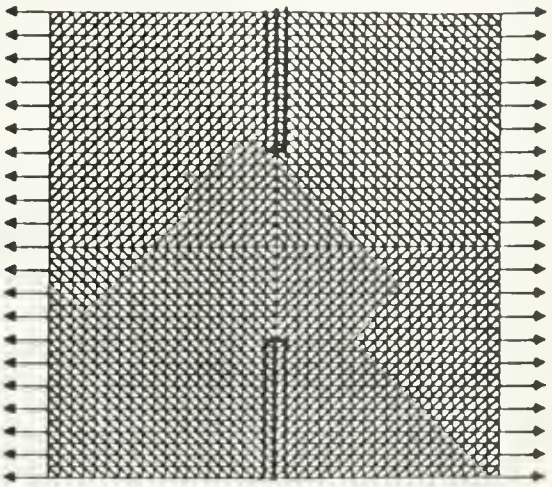


fig8:triangular cuts(shape 3)



a:first discretization



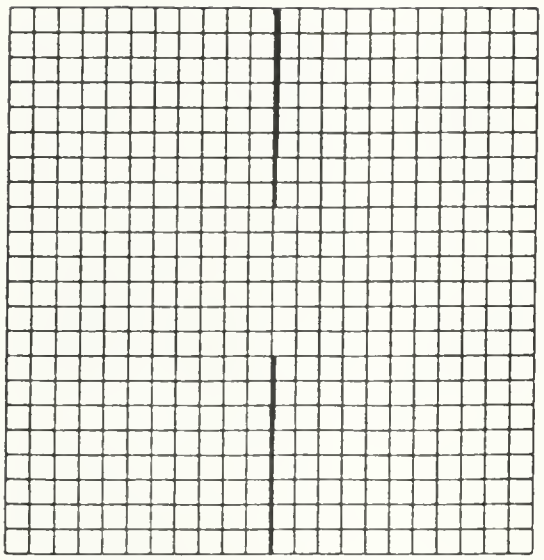
b:second discretization

figure 9 : type of dicretization

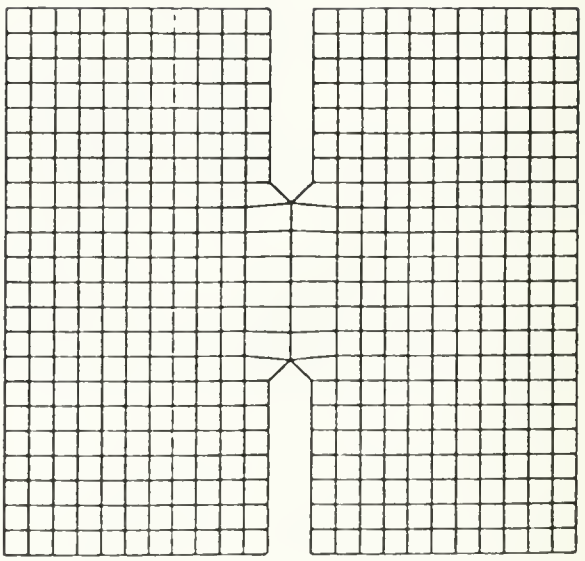
shape 1

shape 2

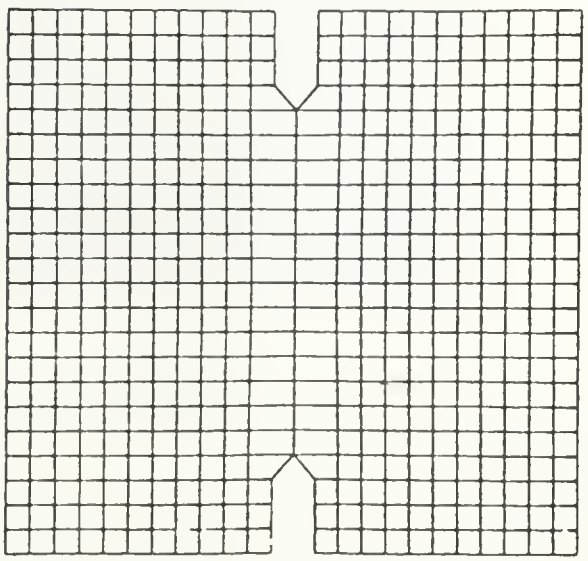
shape 3



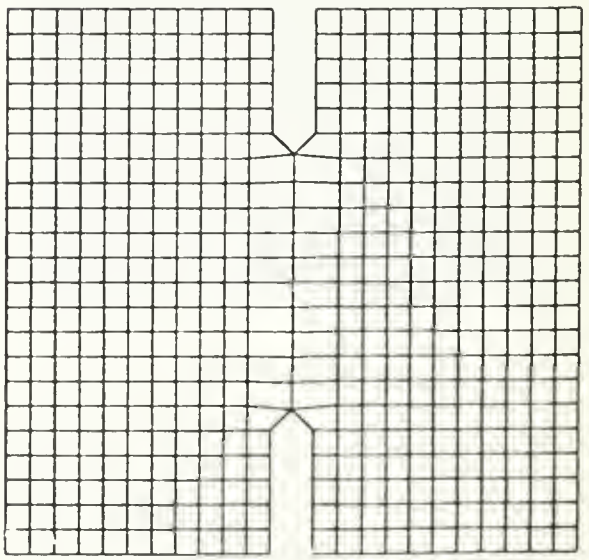
initial structure (l=1/2)



optimal deformation (l=2/3)

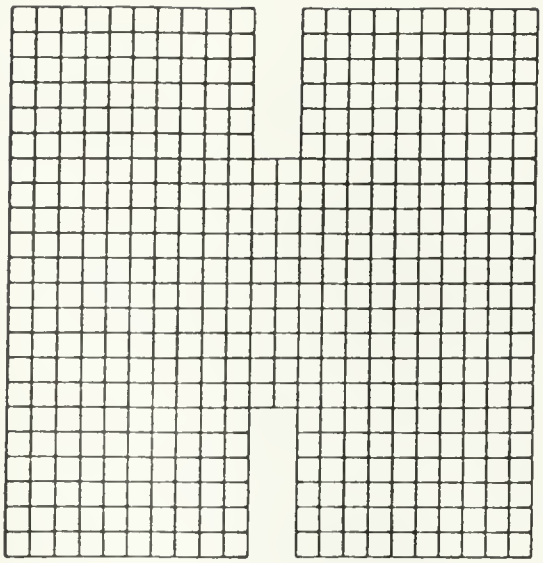


optimal deformation (l=1/3)

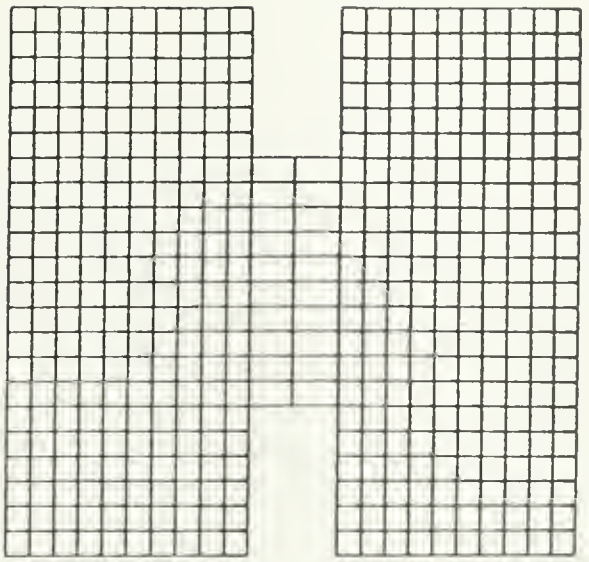


optimal deformation (l=1/2)

figure 10: deformation of the domain (shape 1)

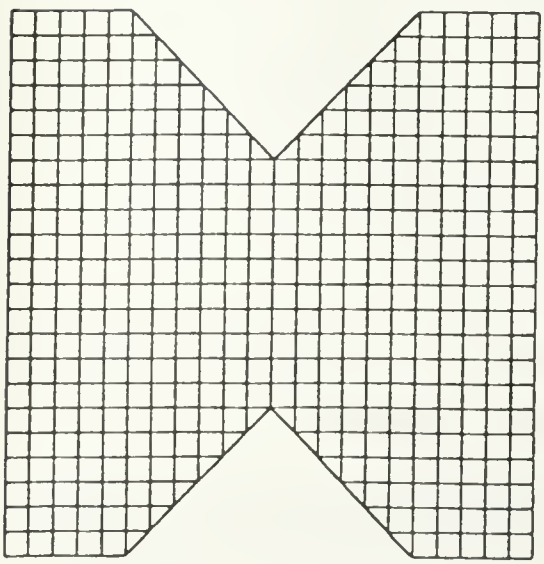


initial structure

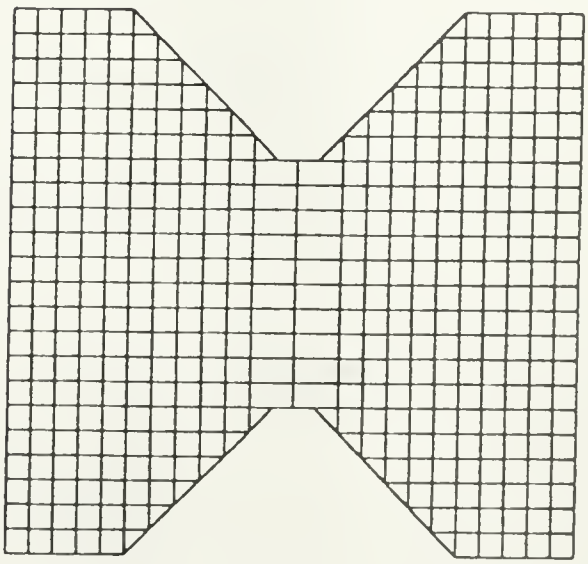


optimal deformation (l=1/2)

figure 11: deformation of the domain (shape 2)



initial structure



optimal deformation (l=1/2)

figure 12: deformation of the domain (shape 3)

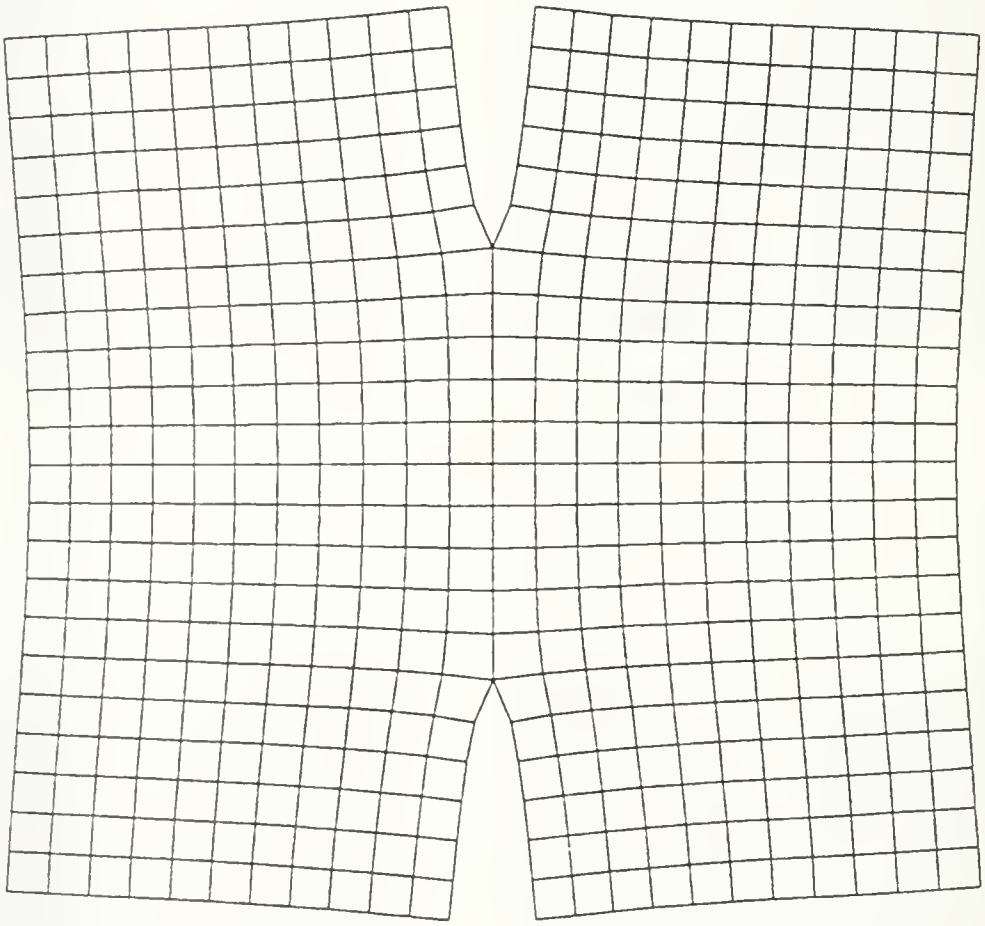


figure 13 : value of the displacement after 13 iterations of the approximate problem.

(shape 1, $l=1/2$, $\varepsilon = 0.1\beta$)
value of the objective function : 0.94775
value of the projected gradient : $3.9\ 10^{-13}$

Table II: Numerical results for the different problems.
$\epsilon_{MCH} = 10^{-16}$, $\epsilon_{ACT} = 5 \times 10^{-4}\beta_N$, $\epsilon_{PG} = 10^{-16}$, $N = 12$

Discre–tization	*Shape of* Ω'	ℓ	*Final value of* f	*Final Value of norm of the gradient*	*No. of Iterations*		*EST*
					Newton	*Steepest Method*	
1	1	1/3	.462064	2.9×10^{-19}	26	26	17%
2	1	1/3	.495431	7.4×10^{-18}	21	5	28%
1	1	1/2	.7191869	2.7×10^{-11}	36	1	23%
2	1	1/2	.752561	0	26	1	35%
1	1	2/3	.9751832	6.9×10^{-1}*	28	9	55%
2	1	2/3	1.009578	1.3×10^{-7}*	32	5	35%
1	2	1/3	.385694	0	22	84	13%
2	2	1/3	.385694	1.3×10^{-18}	19	49	26%
1	2	1/2	.642824	0	26	3	31%
2	2	1/2	.642824	5.2×10^{-18}	23	4	36%
1	2	2/3	.899287	5.7×10^{-15}	45	5	38%
2	2	2/3	.899954	0	21	1	51%
1	3	1/3	.899954	0	28	0	27%
1	3	1/2	1.028518	0	31	0	26%
1	3	2/3	1.157083	0	30	1	45%

(*: the optimization was interrupted because the line search could not find a lower point.)

NYU COMPSCI TR-422
Gaudrat, Veronique F
A Newton type algorithm for
plastic limit analysis.
c.1

DATE DUE	BORROWER'S NAME